三、多路复用IO

2024年6月29日 20:42

多进程服务器

每当有一个新的客户端建立连接，就会创建一个新的进程为这个客户端服务
当某一个客户端断开连接时，子进程终止

问题：

- 频繁的创建进程和销毁进程，系统开销较大
- 能够承载的上限低
- 能否实现一个进程就可以和多个客户端进行通信？（多路IO）

三、多路复用IO

3.1 多路复用IO的概念和作用

多路复用 I/O(Multiplexing I/O)是一种 I/O 模型，用于处理多个 I/O 操作，同时允许程序等待多个输入或输出事件而不会阻塞。它在网络编程中扮演着重要角色，可以提高程序的并发性能和效率

多路复用 I/O 的主要目的是使程序能够同时监听多个文件描述符(通常是套接字)，在有事件发生时立即做出响应，而不需要在等待一个套接字上的 I/O 完成时阻塞整个程序。这种模型有助于避免创建大量线程或进程来处理并发连接，从而节省系统资源并提高程序的性能。

多路复用 1/0 的作用包括:

- 提高并发性能：多路复用 1/0 允许一个线程或进程同时监听多个套接字上的1/0 事件，从而使程序能够同时处理多个连接。
- 减少资源消耗：相比创建大量线程或进程来处理并发连接，多路复用 I/0可以节省系统资源，减少上下文切换的开销。
- 避免阻塞：多路复用 I/0 允许程序在等待 I/0 事件的同时继续执行其他任务，避免了阻塞整个程序。
- 简化编程模型：多路复用 I/0 可以将不同套接字的 I/0 事件汇总到一个地方，简化了编程模型，使代码更加清晰易懂。

常见的多路复用 I/0 模型包括 select、poll、epoll(在 Linux 中)，它们在不同操作系统和环境中具有类似的功能，但可能有不同的性能和用法。多路复用 IO 在服务器编程中经常用于监听多个客户端连接，实现高并发的网络服务。

多路IO复用：几个特殊的函数（select、poll、epoll）

多路IO复用解决了什么问题？

lfd、cfd1、cfd2、cfd3（一个服务器、三个客户端）
他们的读事件什么时候发生，无法确定，因此最开始我们没做任何处理，导致多个客户端连接服务器之后，服务器是读不到消息的。无法实现相关功能。
之后，我们利用多进程（多线程）处理不同的客户端申请连接，但是开销太大。

多路IO：帮助我们在一个进程（线程）下， 一起监听很多个fd的事件（有客户端申请连接或是客户端发送消息），当有事件触发的时候，可以及时的通知用户处理。（多路IO会选择合适的时机去调用accept或者是read，保证不会阻塞）

3.2 select()系统调用

函数描述
对于lfd、cfd1、cfd2
select可以帮助我们监听lfd、cfd1、cfd2的事件，当其中一个或多个事件发生时，select可以立刻通知用户，并告知是那个文件描述符的那个事件发生了。

lfd-->事件触发--->accept()

cfd1-->读事件触发--->read(cfd1)
cfd2-->读事件触发--->read(cfd2)

头文件：

```
#include <sys/select.h>
```

函数原型：

```
int select(int nfds, fd_set *readfds, fd_set *writefds, fd_set *exceptfds, struct timeval *timeout);
```

辅助宏函数

```
void FD_CLR(int fd, fd_set *set);
int  FD_ISSET(int fd, fd_set *set);
void FD_SET(int fd, fd_set *set);
void FD_ZERO(fd_set *set);
```

函数参数：

- int nfds

- fd_set *readfds   // 文件描述符的集合，监听读事件的文件描述符集合
- fd_set *writefds // 文件描述符的集合，监听写事件的文件描述符集合
- fd_set *exceptfds // 文件描述符的集合，监听异常事件的文件描述符集合
- struct timeval *timeout // 超时事件，先传NULL